Analisis *Custom Search* dan *Google Search* pada Sistem Pencarian Data Seminar dan Kompetisi

Theon Leonardus[1], Oscar Wongso, S.Kom., M.T.[2]

Teknik Informatika, Universitas Kristen Maranatha
Jl. Prof. drg. Surya Sumantri No. 65 Bandung
[1]theon.leonarduss@gmail.com
[2]oscar.wongso@it.maranatha.edu

***Abstract* — At this time, the website is a good opportunity to attract people's attention because it can be accessed anywhere and by anyone. As a student, they certainly want to seek more knowledge in terms of whether they are taken or not. And students also mostly seek knowledge with seminars or competitions, by looking for or attending seminars or competitions students can get student points. Therefore, the main function of search engines or search engines is to simplify and speed up search. The background for making this search engine is to help retrieve data that will be searched by students. By doing the scraping method, this will help students in conducting seminar and competition data searches. This site was built with Google API technology applications, including the implementation of Custom Search Engines (CSE), as well as the Document Object Model (DOM) technology application used to create and manipulate data in HTML. Google API will make it easier to create data search applications, by providing custom search engine services from Google. And in this study will discuss the problem of the difference between a custom search engine search engine and google search engine.**

Keywords— **DOM, Google API, Google Custom Search Engine.**

I. PENDAHULUAN

Perkembangan teknologi yang membuat para mahasiswa mencari di sektor informasi, search engine merupakan bagian dari aktivitas sehari-hari. Bagaimanapun mulai dari kalangan akademis ataupun professional, search engine memang sangat dibutuhkan untuk mencari informasi yang cepat dan aktual. Google pertama kali diperkenalkan ke publik pada tahun 1998 oleh Sergey Brin dan Lawrence Page yang merupakan mahasiswa dari Universitas Stanford yang saat itu masih berupa prototype (Brin & Page, 1998). Mesin pencari google terus berevolusi, mulai dari sisi pengembangan algoritma pencarian yang semakin canggih, sampai fitur-fitur baru untuk mempermudah pengguna dalam melakukan pencarian informasi.

Jumlah pengguna internet di Indonesia diproyeksikan tembus 175 juta pada 2019, atau sekitar 65,3% dari total penduduk 268 juta. Peningkatan pengguna internet terutama ditopang oleh semakin meluasnya penggunaan ponsel pintar (smartphone) dan selesainya proyek penggelaran kabel fiber optic Palapa Ring yang menyambungkan jaringan internet ke seluruh wilayah Indonesia. Angka proyeksi tersebut meningkat 32 juta, atau 22,37% dibandingkan survei terakhir Asosiasi Penyelenggara Jasa Internet Indonesia (APJII) tahun 2017 yang mencatat pengguna internet sebanyak 143 jutaan.

Sebagai mahasiswa ingin sekali mendapatkan ilmu di dalam maupun di luar bidang yang sedang mahasiswa tempuh. Terutama untuk mahasiswa berprestasi yang akan berpartisipasi untuk mengikuti kompetisi dalam bidangnya antarkampus. Begitu pula sama halnya dengan mencari informasi seminar yang mahasiswa ingin cari untuk pengetahuan lebih lanjut. Cara pemenuhan kebutuhan akan informasi ini dapat dilakukan dengan beraneka ragam. Mulai dari sekedar membaca koran, majalah, atau jurnal tertulis hingga menggunakan teknologi digital yang terus berkembang.

Lambat laun penggunaan teknologi digital dapat memudahkan dalam mencari informasi. Pembuatan website ini bertujuan untuk membantu mahasiswa dalam pencarian data seminar dan kompetisi, dan juga bisa membantu mahasiswa untuk mengumpulkan point-point kelulusan yang mahasiswa butuhkan. Dan hal ini dibantu dengan menggunakan metode grabbing , yang mana akan mendapatkan sumber data yang ada (Google) dengan cara mengkoneksikan pada aplikasi yang dibuat.

Tujuan yang ingin dicapai dari penelitian ini adalah untuk membuat suatu aplikasi yang dapat membantu pengguna khususnya mahasiswa Fakultas Teknik Informatika dalam pencarian dokumen informasi dengan harapan hasil dari pencarian data dapat dilihat dengan jelas mengenai referensi yang ada.

II. KAJIAN TEORI

Teori yang dijelaskan meliputi : *Framework* CodeIgniter sebagai kerangka pembuatan aplikasi, untuk *knowledge base* pencarian data dengan menggunakan Google API.

A. Google Search Engine

Google *Search Engine* atau disebut mesin pencari Google merupakan salah satu mesin pencari yang dapat mempermudah setiap orang dalam melakukan pencarian informasi di internet, mulai dari kalangan akademisi ataupun profesional mesin pencari web memang sangat dibutuhkan dalam membantu mencari informasi cepat dan aktual, cukup dengan menuliskan satu atau beberapa gabungan kata kunci dari informasi yang akan kita cari, secara otomatis mesin pencari web akan memberikan informasi sesuai dengan kata kunci yang kita masukan.[1]

Sebagian besar mesin pencari dijalankan oleh perusahaan swasta yang menggunakan algoritma kepemilikan dan database tertutup yang paling populer adalah Google. Telah ada beberapa upaya menciptakan mesin pencari dengan sumber terbuka (open-source), contohnya adalah Htdig, Nutch, Egothor dan OpenFTS. Karena masing-masing mesin pencari memiliki algoritma yang berbeda, maka metode yang di gunakan akan di utamakan untuk mesin pencari google.[2]

Search engine merupakan mesin pencari yang berupa sebuah website untuk mencari informasi yang tersimpan di dalam situs yang lain. Tiga tugas dasar sebuah *search engine* dalam cara kerjanya [3] :

1. Mencari di internet atau memilih bagian-bagian dari internet menurut kata-kata penting atau kunci
2. Memberi indeks pada kata-kata yang dicari, dan dimana mereka menemukannya
3. Mengijinkan pengguna untuk mencari kata-kata atau kombinasi kata yang ditemukan pada indeks.

B. Google API

Google API bisa dikatakan bagian dari *Framework* Google. Google menyediakan berbagai API (*Application Programming Interface*) yang sangat berguna bagi pengembang web maupun aplikasi desktop untuk memanfaatkan berbagai fitur yang disediakan oleh Google seperti misalnya: *AdSense*, *Search Engine*, *Translation* maupun *YouTube*. API secara sederhana bisa diartikan sebagai kode program yang merupakan antarmuka atau penghubung antara aplikasi atau web yang kita buat dengan fungsi-fungsi yang dikerjakan. Misalnya dalam hal ini Google API berarti kode program (yang disederhanakan) yang dapat kita tambahkan pada aplikasi atau web untuk mengakses, menjalankan dan memanfaatkan fungsi atau fitur yang disediakan Google. Pada pembuatan aplikasi ini bisa menambahkan fitur Google *Custom Search* pada website. API juga merupakan fungsi fungsi pemrograman yang disediakan oleh aplikasi atau layanan agar layananan tersebut bisa di integrasikan dengan aplikasi yang dibuat. [4]

C. Google Custom Search

Google *Custom Search* merupakan sebuah platform yang disediakan oleh Google, yang memungkinkan pengembang situs untuk menyediakan sebuah mesin pencarian di dalam situs pribadi secara khusus. Sejak 2006, fitur pencarian pada media ini dapat dikonfigurasi berdasarkan batasan-batasan yang ingin ditetapkan oleh pengembang situs. Pengembang dapat menentukan sendiri batasan topik pencarian untuk mengeliminasi situs-situs dengan informasi yang tidak diperlukan. Bahkan pengembang dapat menjadikannya sebagai sumber pendapatan dengan menghubungkan mesin pencarian kepada akun Google *AdSense*. [5]

D. Web Scraping

Web *Scraping* berkaitan erat dengan pengindeksan web, yang indeks konten web menggunakan bot dan merupakan teknik universal yang diadopsi oleh kebanyakan mesin pencari. Sebaliknya, menggores web lebih memfokuskan pada transformasi konten web yang tidak terstruktur, biasanya dalam format HTML, menjadi data terstruktur yang dapat disimpan dan dianalisa dalam database lokal pusat atau spreadsheet. Web Scraping juga terkait dengan otomasi web, yang mensimulasikan browsing web manusia menggunakan perangkat lunak komputer.[6]

Web *scraping* mempunyai sebuah langkah yaitu, **Create Scaping Template**: pembuat program mempelajari dokumen HTML dari website yang akan diambil informasinya untuk tag HTML yang mengapit informasi yang akan diambil. Langkah kedua ***Explore Site Navigation***: Teknik navigasi pada website yang akan diambil informasinya untuk ditirukan pada aplikasi *web scraper* yang akan dibuat. Langkah ketiga ***Automate Navigation and Extraction***: aplikasi web scraper dibuat untuk mengtomasi pengambilan informasi dari website yang ditentukan. Langkah keempat ***Extracted Data and Package History***: informasi yang didapat dari langkah 3 disimpan dalam tabel database.[7]

```
1) DOM dari URL          => $variabel = file_get_contents("$url");
2) Library DOM           => $html = new simple_html_dom();
3) Pengambilan Content   => $variabel_tampung = $html->load($variabel)
                            ->find("tipe_data, parameter);
4) Data disimpan ke dalam data array => $data['variabel_tampung'] = $variabel_tampung;
5) Data array di return => return $data;
```

Gambar 1. Kode Program *Scraping* Struktur Data

Gambar 1 adalah contoh mengenai penggunaan *scraping* digunakan dengan cara mem-*parsing* struktur data yang ingin diambil. Kemudian dibantu dengan bantuan *library "simplehtmldom*" yang mana bertujuan mengambil *content website*. Setelah itu pengambilan struktur data dibantu dengan membuat *Document Object Model* dari URL yang disimpan kedalam sebuah variabel. Kemudian variabel tersebut ditampung ke dalam *object array*.

E. JavaScript

JavaScript adalah bahasa pemograman website yang bersifat CSPL atau *Client Side Programming Language*. *Client Side Programming Language* adalah tipe bahasa pemograman yang pemrosesannya dilakukan oleh *client*. Aplikasi *client* yang dimaksud merujuk kepada web *browser* seperti Google Chrome dan Mozilla Firefox. Jenis bahasa pemograman *Client Side* berbeda dengan bahasa pemograman *Server Side* seperti PHP *(Kurniawan, H. 2011),* dimana untuk *server side* seluruh kode program dijalankan di sisi server. Untuk menjalankan JavaScript, kita hanya membutuhkan aplikasi text editor, dan web browser. JavaScript memiliki fitur: *high-level programming language*, *client-side*, *loosely tiped*, dan berorientasi objek[8]

F. Ajax

Ajax bukan lah merupakan teknologi, dan juga bukan merupakan bahasa pemrograman. Ajax popular semenjak *Garret (2005)* menuliskan artikel berjudul "*Ajax*: *A New Approach to Web Applications*". Ajax dengan memanfaatkan minimal *JavaScript, DOM, dan XMLHttpRequest* dapat digunakan untuk mengurangi kesenjangan antara aplikasi web konvensional dan aplikasi desktop.

Dengan teknik menggunakan AJAX, maka *JavaScript* yang ada pada sebuah halaman web dapat berkomunikasi langsung ke *server*, menggunakan objek *JavaScript XMLHttpRequest*. Dengan objek ini, kode *JavaScript* dapat mengakses data di *server* tanpa harus me-*reload* seluruh halaman web.[9]

G. PHP

PHP adalah bahasa pemrograman *script* yang paling banyak dipakai saat ini. PHP banyak dipakai untuk memrogram situs web dinamis, walaupun tidak tertutup kemungkinan digunakan untuk pemakaian lain. Contoh terkenal dari aplikasi PHP adalah forum (phpBB) dan MediaWiki (software di belakang Wikipedia). PHP juga dapat dilihat sebagai pilihan lain dari ASP.NET/C#/VB.NET Microsoft, ColdFusion Macromedia, JSP/Java Sun Microsystems, dan CGI/Perl. Contoh aplikasi lain yang lebih kompleks berupa CMS yang dibangun menggunakan PHP adalah Mambo, Joomla!, Postnuke, Xaraya, dan lainlain.[10]

H. Framework CodeIgniter

Codeigniter adalah sebuah *framework* untuk *web* yang dibuat dalam format *PHP*. Format ini selanjutnya dapat digunakan untuk membuat sistem aplikasi *web* yang kompleks. *Codeigniter* dapat mempercepat proses pembuatan *web*, karena semua *class* dan *module* yang dibutuhkan sudah ada dan *programmer* hanya tinggal menggunakannya kembali pada aplikasi web yang akan dibuat.[11]

Implementasi metode *model*, *view* dan *controller* dengan *framework Codeigniter*. Sebuah konsep pemrograman yang memisahkan pemrograman *logic* aplikasi dengan presentasinya. 3 jenis komponen yang membangun suatu MVC [12]:

1. Model yaitu, merupakan struktur data. Membantu proses hubungan database.

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class nama_model extends CI_Model {
    public function __construct(){
        $this->load->database();
    }

}
?>
```

Gambar 2. Kode Program Pada Model

2. View, yaitu merupakan informasi yang disampaikan pengguna. *User Interface* dengan melibatkan komponen grafis.
3. Controller, yaitu merupakan sebuah perantara antara *Model* dan *View* dan semua sumber yang dibutuhkan untuk memproses permintaan HTTP.

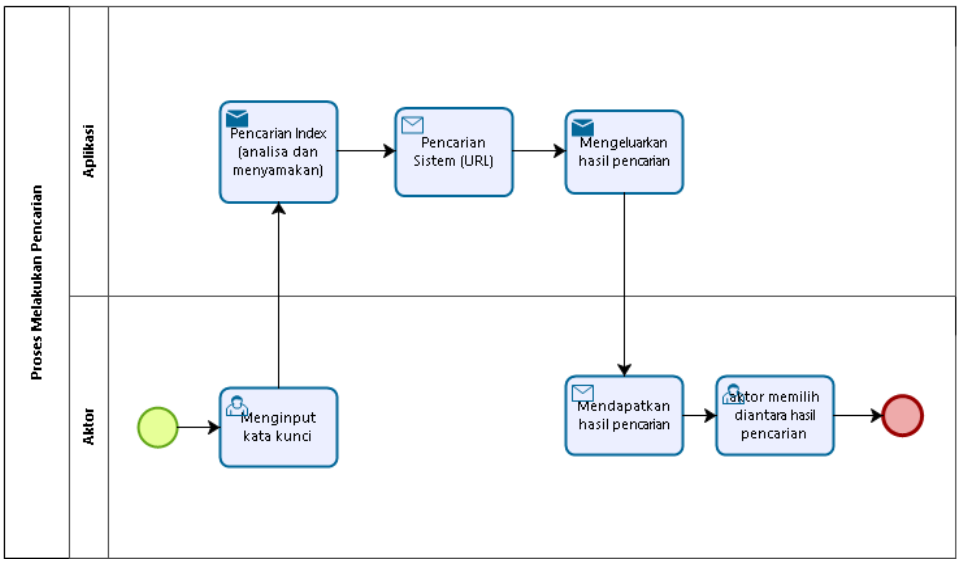
```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
ini_set('max_execution_time', 0);

class nama_controller extends CI_Controller {
    function __construct() {
        parent::__construct();
        include APPPATH . 'third_party/simple_html_dom.php';
        $this->load->model('nama_model');
    }

}
```

Gambar 3. Kode Program Pada Controller

III. ANALISIS DAN RANCANGAN SISTEM

A. Proses Bisnis

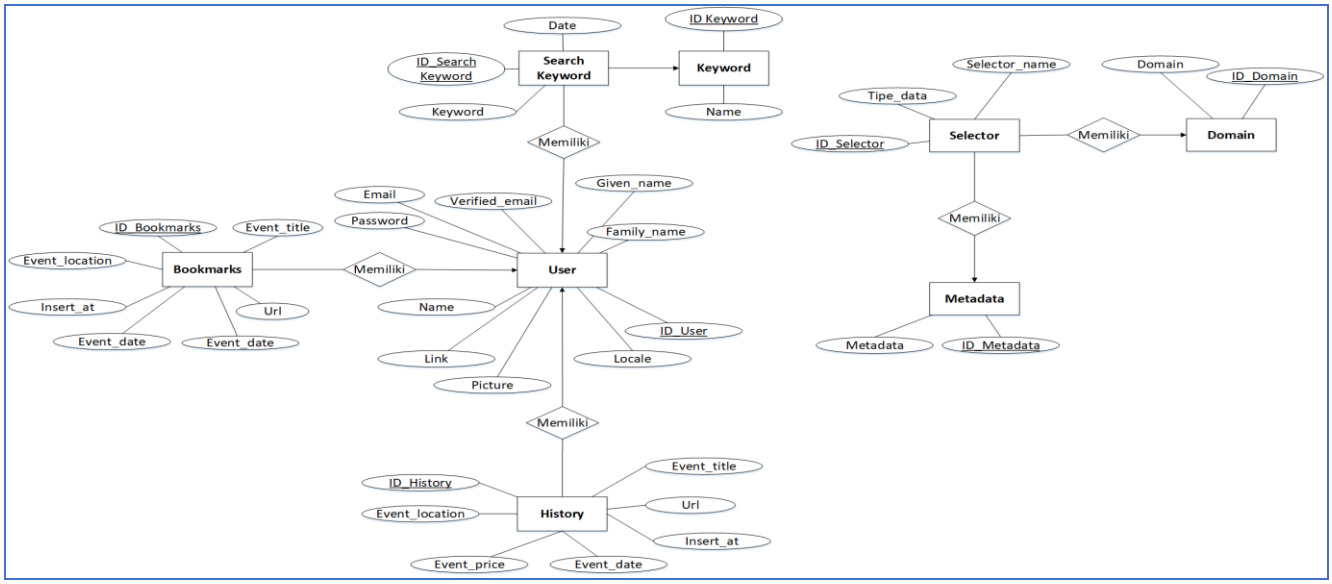
Proses bisnis ini, akan menjelaskan fitur pencarian. Dalam aplikasi pencarian ini sumber yang di dapat terdiri dari 3 *domain website*. *Domain* tersebut terdiri dari "eventbrite.com", "eventkampus.com" dan "dicoding.com". Ketika *user* akan melakukan pencarian dengan menginput berdasarkan kata kunci, maka sistem akan melakukan pengindeksan berdasarkan kata kunci. Lalu akan menghasilkan output pencarian.



Gambar 4. Proses Bisnis *Search Engine*

Pada Gambar 4 diatas menjelaskan bahwa *user* akan melakukan pencarian dengan menginput kata kunci lalu pada aplikasi akan menganalisa atau menyamakan *index* dari kata kunci. Kemudian sistem mencari url yang sama lalu dikeluarkan hasil output pencarian. *User* pun menerima hasil dari pencarian lalu memilih data tersebut.

B. Entity Relationship Diagram



Gambar 5. Entity Relationship Diagram

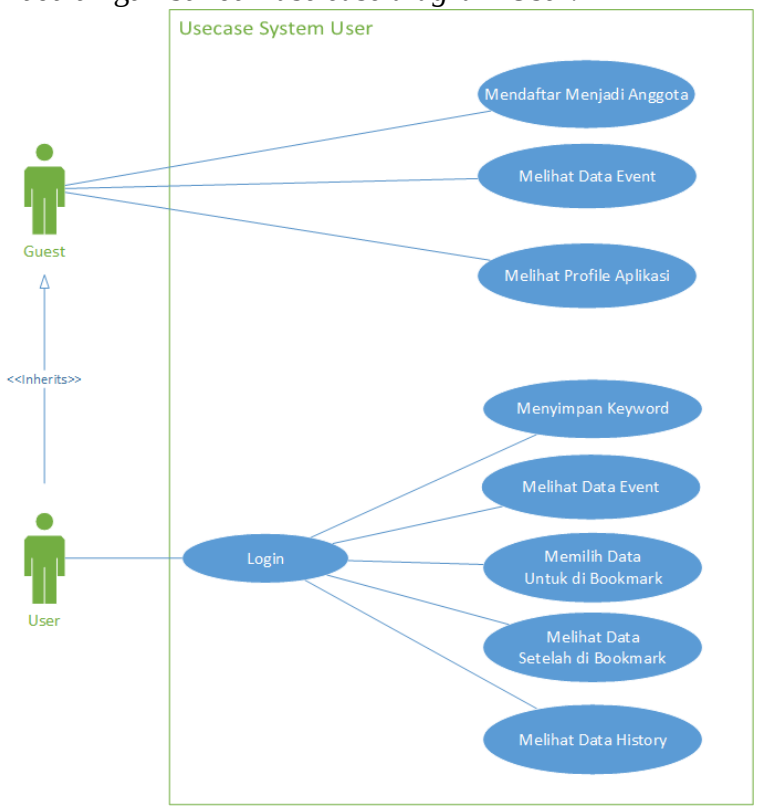
Pada subbab ini, akan menjelaskan perancangan basis data aplikasi yang digambarkan dalam ERD (*Entity Relationship Diagram*) pada Gambar 5 yang memerlukan 8 tabel yaitu tabel *User*, tabel *History*, tabel *Bookmark*, tabel *SearchKeyword*, tabel *Selector*, tabel *Domain*, tabel *Metadata*. Pada tabel *user* akan menyimpan data-data pengguna yang akan memakai program, Tabel *user* memiliki relasi ke tiga tabel yaitu tabel *bookmark*s dan *history*. Tabel *bookmark*s akan menyimpan data bookmarks dari pengguna. Dan tabel *history* akan menyimpan data-data yang telah dikunjungi pengguna. Kemudian tabel *SearchKeyword* akan menyimpan data-data *keyword* yang telah dicari oleh pengguna.

C. Use Case Diagram

Pada subbab ini akan menjelaskan *use case diagram* pada aplikasi. Berikut adalah gambar *use case diagram* pada user:

1) Use Case Diagram User

Berikut dibawah ini adalah gambar dari *use case diagram User*:



Gambar 6. Use Case Diagram *Guest* dan *User*

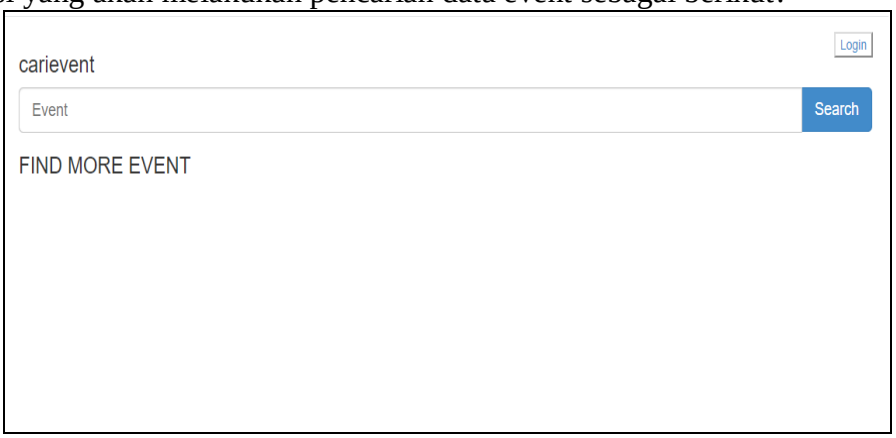
Pada *use case diagram user*, akan menjelaskan bahwa terdapat dua pengguna, yaitu *guest* dan *user* (telah *login*). Seorang *guest* hanya dapat melihat data *event* dan profil aplikasi, kemudian *guest* pun dapat melakukan login. Seorang *user* pada aplikasi sistem pencarian ini dapat melakukan beberapa fitur yang telah diterapkan pada Gambar 6 diatas. *User* akan melakukan *login* terlebih dahulu, kemudian *user* dapat melihat data *event*, menyimpan data untuk di *save*, melihat data yang telah di *save*, melihat data *history* (kunjungan *website*) dan menambah data *event* pada aplikasi.

IV. IMPLEMENTASI

Rancangan antarmuka pada sistem aplikasi ini dirancang untuk pengguna (mahasiswa) yang melakukan pencarian data. Halaman utama hanya menampilkan halaman login dan kotak pencarian yang akan dilakukan oleh pengguna.

A. Halaman Utama Aplikasi

Berikut ini merupakan penjelasan halaman tampilan awal dari website dan fitur-fitur untuk pengguna aplikasi yang akan melakukan pencarian data event sebagai berikut:

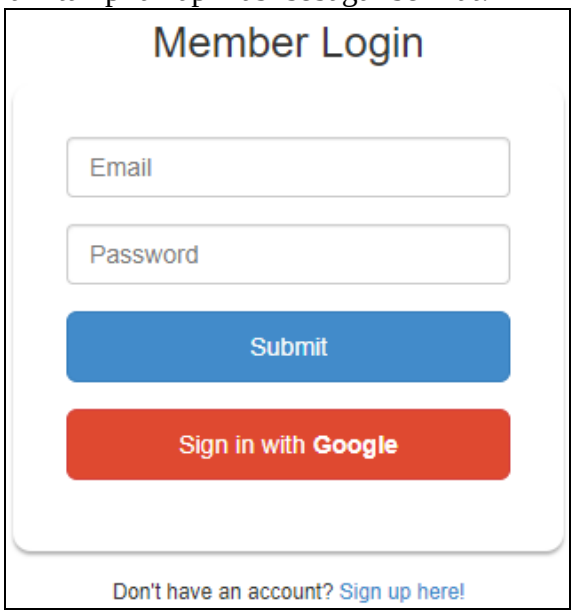


Gambar 7. Halaman Utama Aplikasi

Pada Gambar 7 diatas menunjukan bahwa *user* akan melihat tampilan ini saat akan melakukan pencarian data. Kemudian *user* akan memasukkan kata kunci (*query*) ke dalam *search box* seperti pada gambar di atas. Dan diatas kanan aplikasi terdapat tombol login untuk pengguna masuk kedalam aplikasi. Setelah melakukan login maka pengguna akan mempunyai fitur *user profile*, *keyword* yang tersimpan, *history* hasil web yang dirujuk dan *bookmark* yang akan menyimpan data event.

B. Halaman Login Aplikasi

Berikut ini merupakan penjelasan halaman tampilan login untuk pengguna yang belum mendaftar dari aplikasi dan fitur-fitur dalam tampilan aplikasi sebagai berikut:

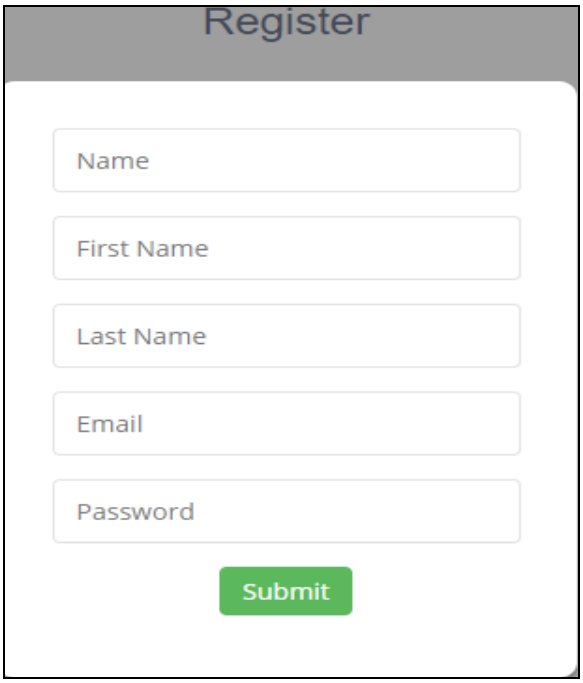


Gambar 8. Implementasi Tampilan Login

Pada Gambar 8 diatas merupakan tampilan login *user*. Halaman ini menjelaskan bahwa *user* dapat melakukan login apabila *user* tidak mempunyai akun Google, maka dapat melakukan registrasi terlebih dahulu dengan meng-klik tulisan *Sign up here*. Lalu *user* hanya akan mengisi alamat *email user* dan *password* yang telah di daftar sebelumnya. Apabila *user* mempunyai akun *gmail* maka dapat melakukan login dengan akun Google.

C. Halaman Registrasi Aplikasi

Berikut ini merupakan penjelasan halaman tampilan registrasi untuk pengguna yang belum mendaftar dari aplikasi dan fitur-fitur dalam tampilan aplikasi sebagai berikut:

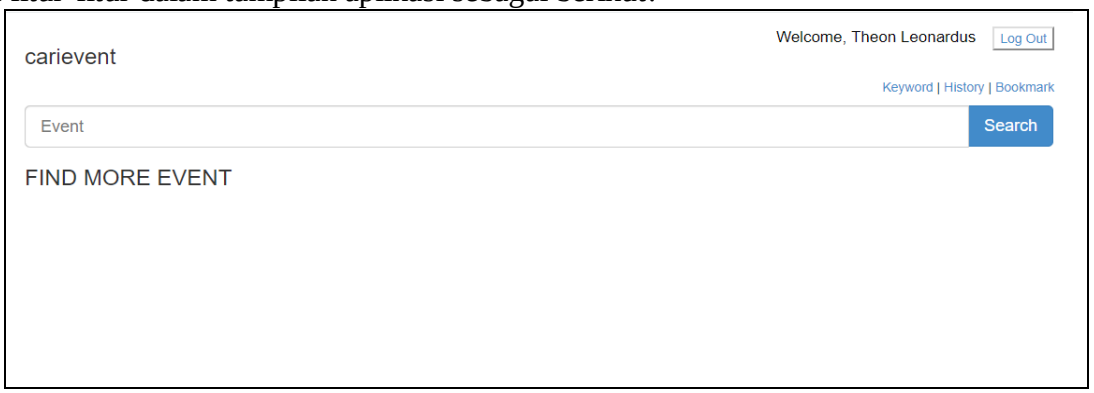


Gambar 9. Implementasi Tampilan Register

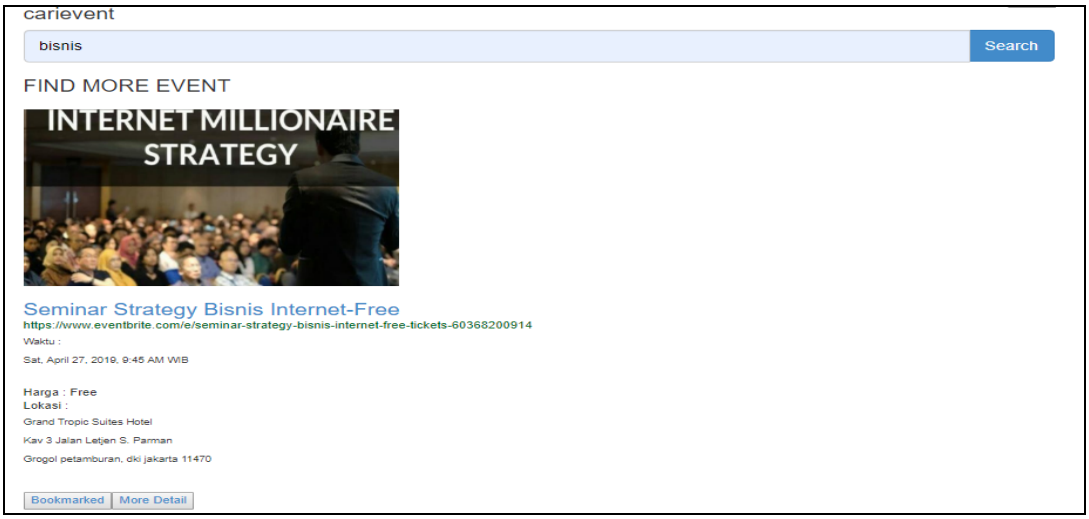
Pada Gambar 9 diatas merupakan tampilan register. Halaman ini menjelaskan apabila *user* tidak mempunyai akun Google, maka dapat melakukan registrasi. Halaman ini *user* diharuskan mengisi *name, first name, last name, email, password*. Setelah itu data *user* akan tersimpan ke dalam *database*.

D. Halaman Hasil Aplikasi

Berikut ini merupakan penjelasan halaman tampilan utama untuk pengguna yang telah melakukan login dan fitur-fitur dalam tampilan aplikasi sebagai berikut:



Gambar 10 Tampilan Utama Setelah Login



Gambar 11. Tampilan Utama Hasil Pencarian

Pada Gambar 11 diatas menunjukan bahwa *user* telah login setelah itu *user* memasukkan kata kunci (*query*) maka akan muncul kemungkinan hasil yang diingkan *user*. Kemudian aplikasi akan menampilkan hasil data *event* dan memunculkan fitur penyimpanan *keyword*, *history* dan *bookmark*.

Pada Gambar 13 diatas menunjukan bahwa setelah *user* memasukkan kata kunci (*query*) maka akan muncul kemungkinan hasil yang diingkan *user*. Pada tampilan ini, hasil yang dikeluarkan akan berupa hasil *scraping* pada suatu *website* dan *url*. Dan terdapat tomblo *bookmark* untuk menyimpan data event.

E. Implementasi AJAX

Proses implementasi terbagi menjadi 3 tahap sesuai dengan rancangan. Tahap pertama yaitu Implementasi AJAX untuk *JavaScript* yang ada pada sebuah halaman *web* dapat berkomunikasi langsung ke *server*, menggunakan objek *JavaScript XMLHttpRequest*. Dengan objek ini, kode *JavaScript* dapat mengakses data di *server* tanpa harus me-*reload* seluruh halaman *web*.

```
$.ajax({
    type: "GET",
    url: url,
    dataType: 'json',
    success: function(result) {
        var data_links = [];
        if(parseInt(result)!=0)
        {

            $.each(result.items, function( i, item ) {
                data_links.push(item.link)
            });
        }
        console.log(data_links);

        // TODO : kirim data Result ke controller untuk diolah
        $('#loadingDiv').removeClass('hide');

        $.ajax({
            type: "POST",
            url: process_data_url,
            dataType: "json",
            data: {'unique_id': unique_id, 'data_links': data_links, 'keyword': query },
            complete: function(jqXHR, textStatus) {
                counter -= 1;
                console.log(counter);
                if (counter <= 0) {
                    $('#loadingDiv').addClass('hide');
                }
            },
            success: function(final_result) {
                console.log("SELESAI");
                counterX -= 1;

                console.log(final_result);

                $.each(final_result.data, function( i, item ) {
```

Gambar 12. Implementasi AJAX

Dalam Gambar 12 diatas, penggunaan AJAX dalam aplikasi terlebih dahulu akan mengambil sebuah data event dari Google, lalu AJAX akan melempar kembali data yang diambil ke *JavaScript* berupa tipe data JSON.

F. Implementasi Metode Scraping

Pada proses implementasi sistem pada situs web ini, ada metode kunci yang menjadi pokok pembahasan, yaitu metode scraping. Metode inilah yang menjadi pokok dari pengembangan situs web, karena melalui metode ini dimungkinkan pengambilan dan pengolahan data serta informasi dari situs-situs berita yang menjadi sumber rujukan berita. Ada kalanya, metode scraping ini tidak dapat dilakukan dikarenakan setiap website tidak memiliki struktur HTML yang sama, dan tidak semua struktur tersebut ditampilkan dalam bentuk *script* melainkan poster. Karena itu, penelitian ini dilakukan dengan pencarian dari berbagai *domain website*.

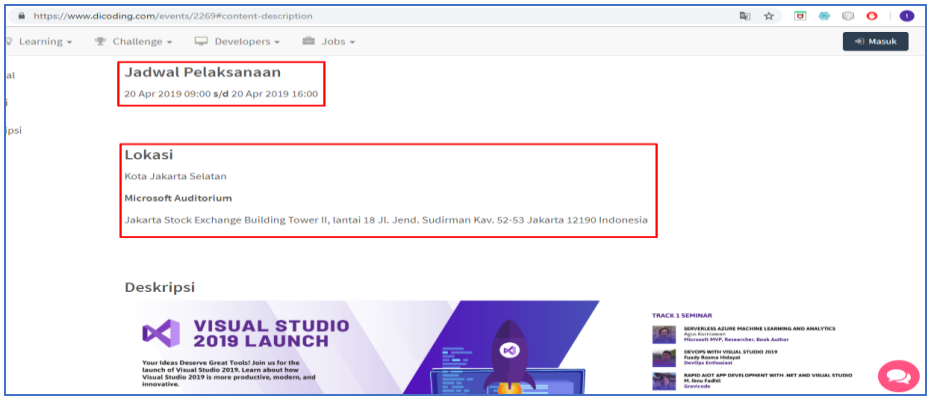
Untuk mendapatkan informasi pada halaman web tersebut harus memuat minimal beberapa element seperti *title*, *date*, *location* dan *price* dari sebuah halaman tersebut. Dapat juga menambahkan *nodes* yang lain, tetapi dalam koteks ini, keempat *node* tersebut sudah cukup mewakili. Dengan demikian, berikut langkah-langkah yang harus dilakukan dalam metode *scraping*.

1. Mengambil seluruh isi/*content* dari laman situs yang dirujuk.
2. Mengambil seluruh *title*, *date*, *location* dan *price* pada laman situs.

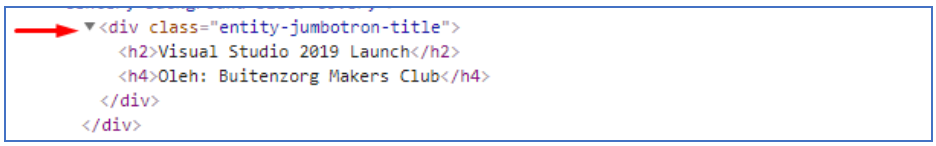
Pengujian akan diukur berdasarkan kesesuaian antara website yang dirujuk yang diambil dengan website yang tersimpan ke dalam basis data. Berikut konteks pengujian yang akan digunakan:

Pertama berjalannya implementasi metode ini dengan baik adalah sumber rujukan berita. Sumber ini yang akan dijadikan benar atau tidaknya implementasi dari metode yang sudah dirancang, beserta struktur *coding* yang dibangun. Dalam konteks ini adalah website berita yang akan dicoba.

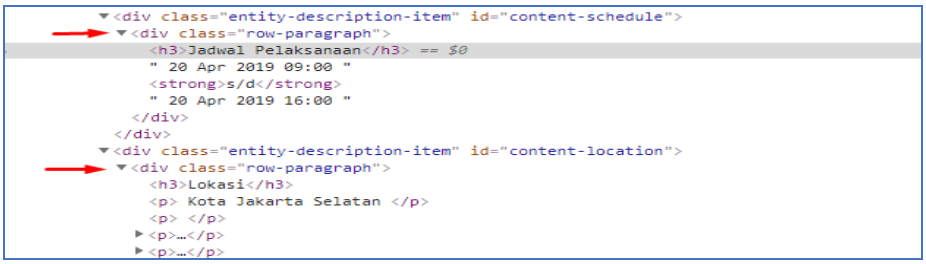
Perhatikan situs berita tersebut pada Gambar 15, sampai Gambar 17 dibawah :



Gambar 13. Laman situs web yang dituju



Gambar 14. Struktur HTML pada laman website



Gambar 15. Struktur HTML pada laman website

Langkah-langkah yang akan mengembangkan struktur *coding* untuk mengimplementasikan metode *scraping* pada situs. Berikut penjabaran potongan-potongan *coding* yang telah dikembangkan.

Pertama, perlu mengambil seluruh isi dari laman yang diinginkan. Terlebih dahulu membuat sebuah *function* bernama *index* untuk menyimpan *coding content*. Hal ini dapat dibantu dengan menggunakan salah satu fungsi yang terdapat pada teknologi PHP, yaitu fungsi "*file_get_content"*. Kemudian seluruh isi laman tersebut dibantu dengan *library "simple_html_dom"* (DOMDocument). *Query* ini yang memungkinkan untuk mengambil seluruh informasi dari *tag* yang inginkan.

Kedua, akan mengambil semua informasi dari *tag* yang butuhkan secara spesifik. Dalam hal ini, informasi yang akan diambil adalah *title*, *date*, *location* dan *price* dari laman website.

Berikut pada Gambar 16 dibawah dalam bentuk *pseudo code* untuk implementasi *scraping*:

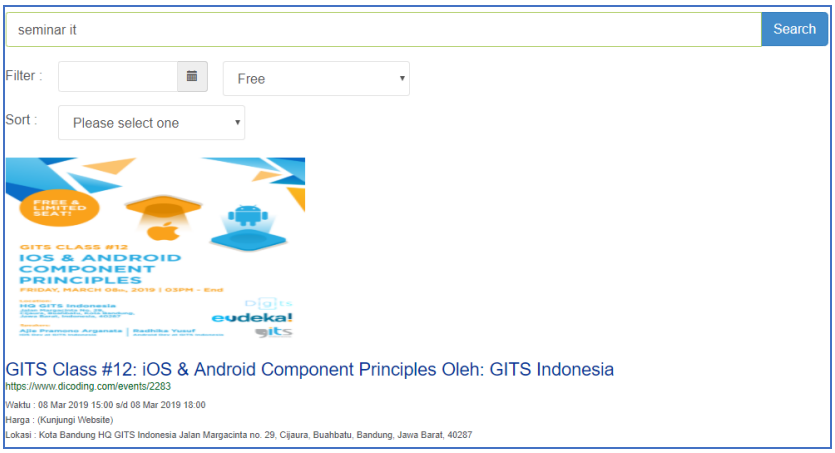
```
function index()
        var kode_html = file_get_content()
        var html = library PHP HTML DOM()

        IF !(event_title)
            RETURN array()
        END IF

        GET all data structur

        var data = data stored in an array

        RETURN data
```

Gambar 16. Implementasi Metode *Scraping*

V. PENGUJIAN

Pengujian ini akan menjelaskan perbedaan ketika melakukan pencarian dalam *Custom Search Engine* dan Google *Search Engine*. Apabila melakukan pencarian dalam *Custom Search Engine* tidak memungkinkan mendapatkan hasil yang sama dalam pencarian Google

1) Uji kasus Pertama



Gambar 17 Hasil Keluaran Kata Kunci "Seminar IT"

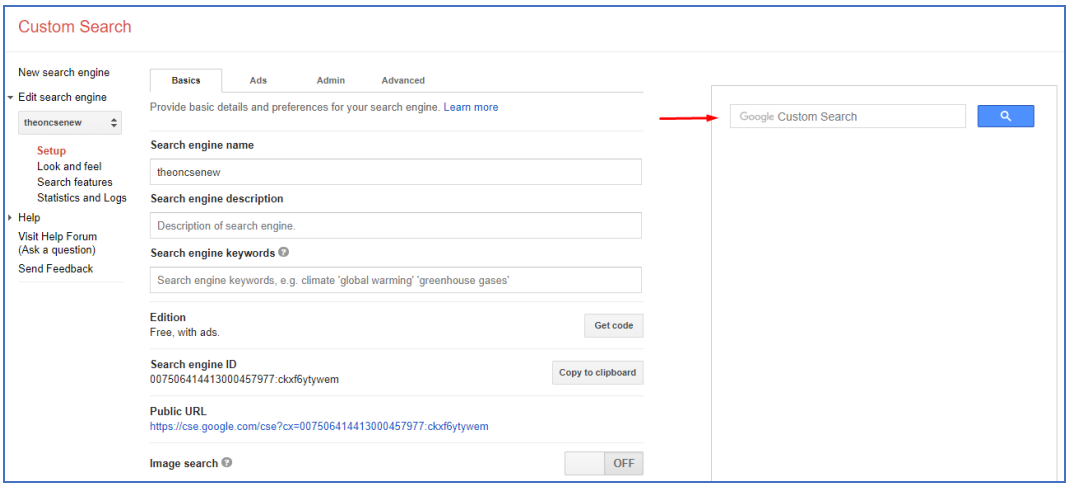
TABEL 1 TABEL ANALISIS KATA KUNCI

Keyword	Output	Simpulan
GITS Class #12: iOS & Android Component Principles Oleh: GITS Indonesia	GITS Class #12: iOS & Android Component Principles Oleh: GITS Indonesia	Sesuai Harapan
GITS Class #12	GITS Class #12: iOS & Android Component Principles Oleh: GITS Indonesia	Tidak Sesuai Harapan
iOS & Android Component Principles	GITS Class #12: iOS & Android Component Principles Oleh: GITS Indonesia	Sesuai Harapan

Pada tabel 1 diatas, menjelaskan untuk melakukan percobaan *keyword* terhadap kesamaan hasil yang dicari berdasarkan *keyword* yang terdapat pada gambar 17. Pada percobaan *keyword* yang kedua berisi "GITS Class #12" memiliki kesimpulan "tidak sesuai harapan" dikarenakan Google menghasilkan pelengkapan otomatis berdasarkan kueri populer untuk mesin pencari. Jika mengatur *search engine* untuk mencari seluruh web, pelengkapan otomatis selain pelengkapan otomatis yang dibuat yang ditambahkan sendiri mungkin tidak tersedia untuk ditampilkan. Kemudian masalah lain dari kueri yang popular, keyword tersebut tidak muncul karena terdapat perbedaan antara *Custom Search Engine* dan *Google Search Engine*. Penjelasan ini dapat dilihat dalam pengujian berikut:

2) Uji Kasus Kedua

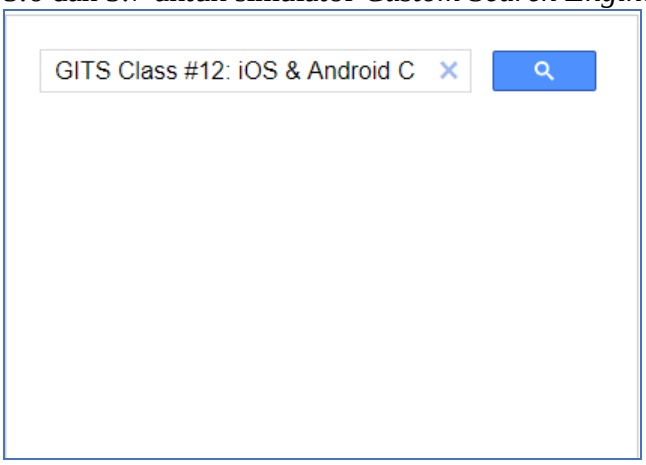
Pengujian menggunakan *Console Custom Search Engine*. Pada pengujian ini akan menguji kata kunci yang terdapat pada tabel 5.3 diatas, untuk hasil dari *Custom Search* sendiri.



Gambar 18. Custom Search Engine

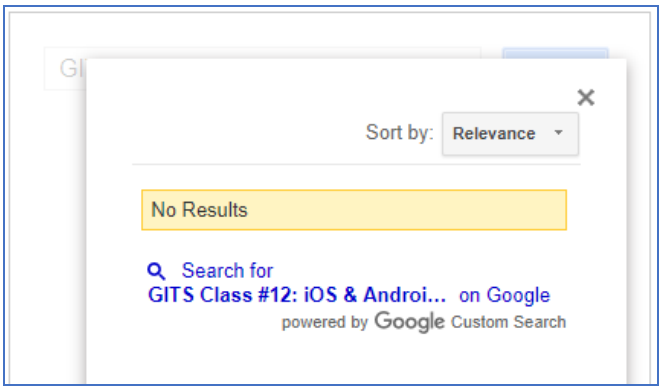
Pada Gambar 18 merupakan tampilan dari *Custom Search Engine* yang telah mempunyai project *search engine*. Pada kolom pencarian yang terletak di kanan merupakan simulator untuk pencarian melalui *Custom Search* ini.

Perhatikan gambar 5.6 dan 5.7 untuk simulator *Custom Search Engine* dibawah:



Gambar 19. Pencarian Melalui Custom Search Engine

Pada Gambar 19 merupakan pengujian pencarian kata kunci melalui *CSE*. Apabila *keyword* pada tabel 1 dimasukkan. Maka tidak akan keluar hasil, dikarenakan peneliti tidak mengkonfigurasi *Custom Search Engine* untuk mencari ke semua web.



Gambar 20 Simulator Custom Search Engine

Pada Gambar 20 menjelaskan bahwa hasil yang dicari tidak ditemukan di simulator *Custom Search Engine*. Dan diketahui juga, pada simulator bahwa hasil dari kata kunci tersebut dapat dicari di Google *Search Engine*.

Apabila pencarian tidak melihat hasil tertentu yang diharapkan maka yang pertama harus dilihat adalah pastikan telah memasukkan hasil atau pola yang cocok dengan hasil di *Custom Search Engine* yang dibuat. Waspada terhadap kesalahan pengejaan dan pastikan halaman yang diharapkan ada di indeks Google. Cara sederhana untuk memeriksa adalah melakukan pencarian di Google dengan situs: operator bersama dengan beberapa kata pada halaman itu sebagai permintaan pencarian.

VI. KESIMPULAN

Ada permasalahan yang ditemukan pada penelitian ini, yaitu tidak munculnya hasil pencarian pada website setelah dilakukan proses *scraping*, akan tetapi hasil pencarian tersebut muncul, ketika proses pencarian tersebut dilakukan dengan menggunakan Google *Search Engine*.

Permasalahan ini terjadi ketika terdapat perbedaan proses pencarian dengan menggunakan *keyword* yang berbeda, yang di mana ketika *user* menggunakan *keyword* yang dilakukan oleh *Custom Search Engine* dengan hasil pencarian pada Google. Hal ini diakibatkan oleh beberapa hal, yaitu terkait kepopuleran, *search engine optimation* seperti pengujian pada tabel 1. *Custom Search* pun mengeluarkan hasil pencarian yang dapat dilihat melalui *content*/isi website, jadi hasil yang dicari tidak memungkinkan mengeluarkan hasil yang cocok dengan yang dikembalikan oleh Google *Web Search*.

Custom Search dikonfigurasikan untuk menelusuri seluruh web, tetapi yang dikonfigurasi dirancang untuk lebih menekankan kepada hasil dari situs domain eventbrite.com, eventkampus.com dan dicoding.com. Dan terdapat kesimpulan bahwa judul yang dimasukkan ke dalam pencarian sebagai *keyword*, tidak memungkinkan untuk mengeluarkan hasil yang sama. Kata kunci untuk *Custom Search Engine* ini harus menggambarkan konten atau subjek *search engine* yang dibuat.

"Jika menggunakan kata kunci dan tidak melihat semua halaman yang diharapkan muncul, coba hapus beberapa dari kata kunci. Menggunakan terlalu banyak kata kunci dapat membatasi hasil yang tidak optimal." Dan memungkinkan hasil yang dicari tidak memiliki kepopuleran atau banyak saingan terhadap website lain.[13]

DAFTAR PUSTAKA

[1] Kurniadi, D., & Mulyani, A., "Pengaruh Teknologi Mesin Pencari Google Terhadap Perkembangan Budaya dan Etika Mahasiswa", pp 19–25, 2016.
[2] Risyad, I., & Maulina, D, "Pencegahan Timbulnya Dork pada Search Eengine Google dengan Plugin", vol. 20, pp 42–47, 2016.
[3] Juliasari, N., & Sitompul, J. C, "Aplikasi Search Engine dengan Metode Depth First Search" (DFS), vol. 9, pp 9–12, 2012.
[4] Edusainstek, S. N., Danang, D., Febriyantahanuji, F., "Rancang Bngun Sistem Informasi Pariswisata dan Budaya Berbasis Web Menggunakan Google API pada Kantor FMIPA UNIMUS", pp 232–241, 2018.
[5] "Google," 16 April 2019. [Online]. Available : https://developers.google.com/custom-search/ [Accessed 16 May 2019]
[6] Utomo, M. S. "Web Scraping pada Situs Wikipedia menggunakan Metode Ekspresi Regular", 18(2), pp 153–160, 2013.

[7] Rifqi, M., “Integrasi Laman Web Tentang Pariwisata Daerah Istimewa Yogyakarta Memanfaatkan Teknologi Web Scraping dan Text Mining”, pp 71–80.
[8] Map, S. C., Peta, P., Sosial, D., Semarang, K., & Blogger, B. “Jurnal Geodesi Undip”, pp 117–130, 2015.
[9] Salatiga, J. D., & Web, P. T., Penggunaan AJAX pada Pengembangan Aplikasi Web, 4(1), pp 86–100, 2007.
[10] Ramadhani, S., Anis, U., & Masruro, S. T., “Rancang Bangun Sistem Informasi Geografis Layanan Kesehatan Di Kecamatan Lamongan Dengan PHP MySQL”, 5(2), pp 479–484, 2013.
[11] Prabowo, D., “Website E-COMMERCE Menggunakan Model View Controller (MVC) dengan FRAMEWORK CODEIGNITER Studi Kasus : Toko Miniatur Pendahuluan Landasan Teori, 16(1), pp 23–29, 2015.
[12] Tanjung, M. Analisis dan Perancangan Sistem Informasi Berbasis Website Menggunakan Arsitektur MVC dengan FRAMEWORK CODEIGNITER Studi Kasus : Ikatan Pelajar Mahasiswa Kepulauan Riau Yogyakarta.
[13] "Support Google," 21 May 2019. [Online]. Available: https://support.google.com/customsearch/. [Accessed 21 May 2019]